University of Central Punjab

Faculty of Information Technology

Data Structures and Algorithms
Spring 2024

Lab 03	
Topic	• Abstract Classes • Templates • Stacks • Stack Application
Objective	The basic purpose of this lab is to implement ADT of stack, and test its applications.

Instructions:

- Indent your code.
- Comment your code.
- Use meaningful variable names.
- Plan your code carefully on a piece of paper before you implement it.
- Name of the program should be same as the task name. i.e. the first program should be Task_1.cpp
- **void main() is not allowed. Use int main()**
- **You have to work in multiple files. i.e separate .h and .cpp files**
- **You are not allowed to use system("pause")**
- **You are not allowed to use any built-in functions**
- **You are required to follow the naming conventions as follow:**
    - **Variables:** firstName; (no underscores allowed)
    - **Function:** getName(); (no underscores allowed)
    - **ClassName:** BankAccount (no underscores allowed)

Students are required to complete the following tasks in lab timings.

Task 1

Create a C++ generic abstract class named as **Stack**, with the following:

Attributes:

1. Type * stackArray;
2. int maxSize;
3. int stackTop;

Functions:

virtual void Push(Type) = 0;

- Should add the element at the top of

stack virtual Type Pop() = 0;

- Should remove the element from the top of stack

Task 2

Stack:

Stacks are a type of container adaptors with LIFO (Last in First Out) type of working, where a new element is added at one end and (top) an element is removed from that end only. So, use the class made in task 1 to make a class named as my**Stack**, having following additional functionalities:

bool empty() : Returns whether the **Stack** is empty or not. Complexity should be: O(1)

bool full() : Returns whether the **Stack** is full or not. Complexity should be: O(1)

int size() : Returns the current size of the **Stack**. Complexity should be: O(1)

Type top () : Returns the last element of the **Stack.** Time Complexity should be: O(1)

Implement both pure virtual functions Push () and pop() declared in base in myStack

After Implementation of the functions in myStack create menu based program to perform the following operations .:

1. Press 1 to add a new item to the stack. **void push(Type)**
2. Press 2 to remove and return the last element from the stack. **Type pop()**
3. Press 3 to check if the stack is full. **bool full()**
4. Press 4 to check if the stack is empty. **bool empty()**
5. Press 5 to return the size of the stack. **int size()**
6. Press 6 to display the stack.
7. Press 0 to exit.

- Write non-parameterized constructor for the above class.
- Write Copy constructor for the above class.
- Write Destructor for the above class.

Task 3

Stack:

After Implementation of the functions in myStack create menu based program to perform the following operations

Make changes to the stack created in Task 2 according to the following. .:

1. Press 1 to add a new item to the stack. **void push(Type)**
2. Press 2 to remove and return the last element from the stack. **Type pop()**
3. Press 3 to check if the stack is full. **bool full()**
4. Press 4 to check if the stack is empty. **bool empty()**
5. Press 5 to return the size of the stack. **int size()**
6. Press 6 to display the stack.
7. Press 7 to count even and odd numbers in the stack
8. Press 8 to count sum of even numbers present in the stack
9. Press 9 to count sum of odd numbers present in the stack
10. Press 0 to exit.